

Reprezentacja wiedzy w indukcyjnym programowaniu logicznym

Agnieszka Ławrynowicz

Instytut Informatyki
Politechnika Poznańska

marzec 2006

Plan prezentacji

Plan prezentacji

1 Wprowadzenie

Plan prezentacji

1 Wprowadzenie

- *reprezentacja wiedzy, pojęcie ontologii*

Plan prezentacji

- 1 Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- 2 Logika deskrypcyjna

Plan prezentacji

- ① Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- ② Logika deskrypcyjna
 - *język reprezentacji wiedzy*

Plan prezentacji

- 1 Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- 2 Logika deskrypcyjna
 - *język reprezentacji wiedzy*
- 3 Indukcyjne programowanie logiczne

Plan prezentacji

① Wprowadzenie

- *reprezentacja wiedzy, pojęcie ontologii*

② Logika deskrypcyjna

- *język reprezentacji wiedzy*

③ Indukcyjne programowanie logiczne

- *jak można wykorzystać wiedzę dziedzinową do indukcji nowej wiedzy*

Plan prezentacji

- 1 Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- 2 Logika deskrypcyjna
 - *język reprezentacji wiedzy*
- 3 Indukcyjne programowanie logiczne
 - *jak można wykorzystać wiedzę dziedzinową do indukcji nowej wiedzy*
- 4 Hybrydowe języki reprezentacji wiedzy oparte na logice

Plan prezentacji

- ❶ Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- ❷ Logika deskrypcyjna
 - *język reprezentacji wiedzy*
- ❸ Indukcyjne programowanie logiczne
 - *jak można wykorzystać wiedzę dziedzinową do indukcji nowej wiedzy*
- ❹ Hybrydowe języki reprezentacji wiedzy oparte na logice
 - *łącznie logikę deskrypcyjną i "programowanie logiczne"*

Plan prezentacji

- 1 Wprowadzenie
 - *reprezentacja wiedzy, pojęcie ontologii*
- 2 Logika deskrypcyjna
 - *język reprezentacji wiedzy*
- 3 Indukcyjne programowanie logiczne
 - *jak można wykorzystać wiedzę dziedzinową do indukcji nowej wiedzy*
- 4 Hybrydowe języki reprezentacji wiedzy oparte na logice
 - *łącznie logikę deskrypcyjną i "programowanie logiczne"*
- 5 Cel i zakres badań

Ontologia z punktu widzenia filozofa

Definicja

Ontologia (**metafizyka**) to dział filozofii zajmujący się teorią bytu

Ontologia z punktu widzenia filozofa

Definicja

Ontologia (**metafizyka**) to dział filozofii zajmujący się teorią bytu

- *nauka o bycie* (Arystoteles, *Metafizyka*, ks. IV)

Ontologia z punktu widzenia filozofa

Definicja

Ontologia (**metafizyka**) to dział filozofii zajmujący się teorią bytu

- *nauka o bycie* (Arystoteles, *Metafizyka*, ks. IV)
- zajmuje się dociekaniem natury "wszystkiego co jest", zarówno bytów rzeczywistych (przedmioty, zdarzenia), jak i abstrakcyjnych (pojęcia, kategorie, terminy)

Ontologia z punktu widzenia filozofa

Definicja

Ontologia (**metafizyka**) to dział filozofii zajmujący się teorią bytu

- *nauka o bycie* (Arystoteles, *Metafizyka*, ks. IV)
- zajmuje się dociekaniem natury "*wszystkiego co jest*", zarówno bytów rzeczywistych (przedmioty, zdarzenia), jak i abstrakcyjnych (pojęcia, kategorie, terminy)
- próbuje odpowiedzieć na pytania takie jak:
 - *co to jest byt?*
 - *co charakteryzuje byt?*
 - *jakie są powiązania pomiędzy bytami?*
 - *jak dokonywać klasyfikacji bytów?*

Ontologia z punktu widzenia informatyka

Definicja

“explicit specification of a conceptualization” [Gruber 93]

“shared understanding of some domain of interest” [Uschold & Gruninger 96]

Ontologia z punktu widzenia informatyka

Definicja

“explicit specification of a conceptualization” [Gruber 93]

“shared understanding of some domain of interest” [Uschold & Gruninger 96]

- Ontologia to innymi słowy **formalna specyfikacja jakiejś dziedziny**, która ma pomóc we współdzieleniu wiedzy oraz w jej wielokrotnym użytku (poprzez oddzielenie **wiedzy dziedzinowej** od wiedzy operacyjnej)

Ontologia z punktu widzenia informatyka

Definicja

“explicit specification of a conceptualization” [Gruber 93]

“shared understanding of some domain of interest” [Uschold & Gruninger 96]

- Ontologia to innymi słowy **formalna specyfikacja jakiejś dziedziny**, która ma pomóc we współdzieleniu wiedzy oraz w jej wielokrotnym użytku (poprzez oddzielenie **wiedzy dziedzicznej** od wiedzy operacyjnej)
- Elementy składowe ontologii:
 - słownik (pojęcia występujące w ramach danej dziedziny),
 - specyfikacja znaczenia elementów słownika (struktura, hierarchia pojęć, relacje między pojęciami) oraz
 - zbiór ograniczeń modelujących dodatkową wiedzę na temat dziedziny

Języki do reprezentacji ontologii

Przykłady języków:

- **Reprezentacja graficzna**: sieci semantyczne, *Topic Maps*, UML, RDF
- **Reprezentacja oparta na logice**: logika deskrypcyjna, reguły logiczne (programowanie logiczne/Prolog, RuleML), logika pierwszego rzędu, frame logic, logiki niemonotoniczne, ...
- **Metody probabilistyczne** np. sieci bayesowskie

Języki do reprezentacji ontologii

Przykłady języków:

- **Reprezentacja graficzna**: sieci semantyczne, *Topic Maps*, UML, RDF
 - **Reprezentacja oparta na logice**: logika deskrypcyjna, reguły logiczne (programowanie logiczne/Prolog, RuleML), logika pierwszego rzędu, frame logic, logiki niemonotoniczne, ...
 - **Metody probabilistyczne** np. sieci bayesowskie
-
- Potrzebny język, w którym można tworzyć definicje pojęć zrozumiałe dla **ludzi** i **maszyn**

Języki do reprezentacji ontologii

Przykłady języków:

- **Reprezentacja graficzna**: sieci semantyczne, *Topic Maps*, UML, RDF
 - **Reprezentacja oparta na logice**: logika deskrypcyjna, reguły logiczne (programowanie logiczne/Prolog, RuleML), logika pierwszego rzędu, frame logic, logiki niemonotoniczne, ...
 - **Metody probabilistyczne** np. sieci bayesowskie
-
- Potrzebny język, w którym można tworzyć definicje pojęć zrozumiałe dla **ludzi i maszyn**
 - Im bardziej **sformalizowany** język tym łatwiejszy do **maszynowego przetwarzania** (np. automatycznego wnioskowania)

Logika deskrypcyjna

Definicja

Definicja

Logika deskrypcyjna (ang. *Description Logics*, \mathcal{DL}) = rodzina opartych na logice formalizmów służących do reprezentacji wiedzy, odpowiednich szczególnie do reprezentacji i wnioskowania o wiedzy terminologicznej (konceptyjnej), ontologiach.

Definicja

Definicja

Logika deskrypcyjna (ang. *Description Logics*, \mathcal{DL}) = rodzina opartych na logice formalizmów służących do reprezentacji wiedzy, odpowiednich szczególnie do reprezentacji i wnioskowania o wiedzy terminologicznej (konceptyjnej), ontologiach.

- podzbiór logiki pierwszego rzędu (**rozstrzygalność**, **wydajność**, **ekspresyjność**)
- formalna semantyka
- korzenie: sieci semantyczne, ramy

Elementy składowe \mathcal{DL}

- **pojęcia** (ang. concepts)
- **role** (ang. roles)
- **konstruktory** (ang. constructors)
- **obiekty** (ang. individuals)

Przykłady

Atomowe pojęcia: **Artist, Movie**

Atomowa rola: **creates**

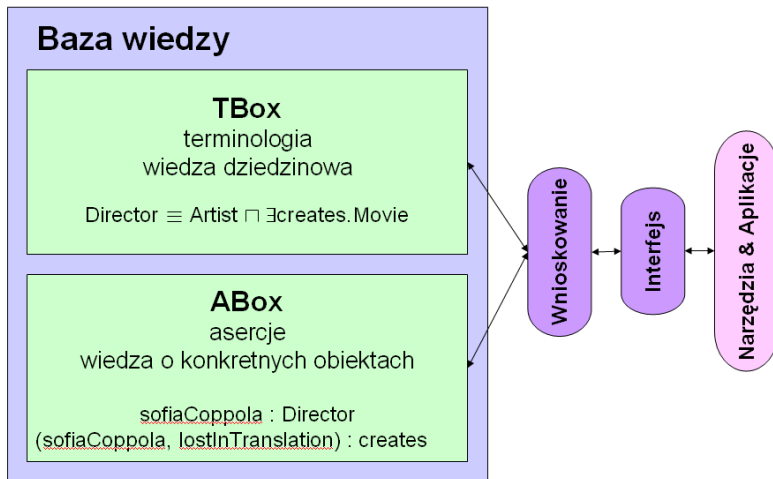
Konstruktory: \sqcap, \exists

Definicja pojęcia: **Director** \equiv **Artist** $\sqcap \exists \text{creates.Movie}$

Aksjomat ("każdy reżyser to artysta"): **Director** \sqsubseteq **Artist**

Asercja: (**sofiaCoppola, lostInTranslation**) : **creates**

Architektura bazy wiedzy w \mathcal{DL}



Wnioskowanie:

- **TBox:** Subsumption, Satisfiability, Classification
- **ABox:** Instance Checking, Consistency

Rodzina języków \mathcal{DL}

- różne zbiory konstruktorów prowadzą do różnych języków

Rodzina języków \mathcal{DL}

- różne **zbiory konstruktorów** prowadzą do **różnych języków**
- najmniejszy język, interesujący z praktycznego punktu widzenia to \mathcal{AL}
 - pojęcia budowane za pomocą konstruktorów: $\sqcap, \neg, \exists, \forall$

Rodzina języków \mathcal{DL}

- różne **zbiory konstruktorów** prowadzą do **różnych języków**
- najmniejszy język, interesujący z praktycznego punktu widzenia to \mathcal{AL}
 - pojęcia budowane za pomocą konstruktorów: $\sqcap, \neg, \exists, \forall$
- dodatkowe litery w nazwie języka oznaczają kolejne rozszerzenia, np.:
 - \mathcal{C} - negacja dowolnych pojęć (nie tylko atomowych)
 - \mathcal{S} - zamiast \mathcal{ALC} z rolami przechodnimi (R^+)
 - \mathcal{O} - singletony (w formie $\{x\}$)
 - \mathcal{I} - role odwrotne
 - \mathcal{N} - ograniczenia ilościowe (w formie $\geq n\mathcal{R}, \leq n\mathcal{R}$)

Rodzina języków \mathcal{DL}

- różne **zbiory konstruktorów** prowadzą do **różnych języków**
- najmniejszy język, interesujący z praktycznego punktu widzenia to \mathcal{AL}
 - pojęcia budowane za pomocą konstruktorów: $\sqcap, \neg, \exists, \forall$
- dodatkowe litery w nazwie języka oznaczają kolejne rozszerzenia, np.:
 - \mathcal{C} - negacja dowolnych pojęć (nie tylko atomowych)
 - \mathcal{S} - zamiast \mathcal{ALC} z rolami przechodnimi (R^+)
 - \mathcal{O} - singletony (w formacie $\{x\}$)
 - \mathcal{I} - role odwrotne
 - \mathcal{N} - ograniczenia ilościowe (w formacie $\geq nR, \leq nR$)
- rozszerzenia w wielu kierunkach: konkretne dziedziny, punkty stałe, operatory epistemiczne, ...

Rodzina języków \mathcal{DL}

- różne **zbiory konstruktorów** prowadzą do **różnych języków**
- najmniejszy język, interesujący z praktycznego punktu widzenia to \mathcal{AL}
 - pojęcia budowane za pomocą konstruktorów: $\sqcap, \neg, \exists, \forall$
- dodatkowe litery w nazwie języka oznaczają kolejne rozszerzenia, np.:
 - \mathcal{C} - negacja dowolnych pojęć (nie tylko atomowych)
 - \mathcal{S} - zamiast \mathcal{ALC} z rolami przechodnimi (R^+)
 - \mathcal{O} - singletony (w formacie $\{x\}$)
 - \mathcal{I} - role odwrotne
 - \mathcal{N} - ograniczenia ilościowe (w formacie $\geq n\mathcal{R}, \leq n\mathcal{R}$)
- rozszerzenia w wielu kierunkach: konkretne dziedziny, punkty stałe, operatory epistemiczne, ...
- im większa **ekspresyjność** języka tym większa **złożoność** wnioskowania

Semantyka \mathcal{DL}

- **Interpretacja \mathcal{I}** $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, gdzie:
 - $\Delta^{\mathcal{I}}$ jest niepustym zbiorem (**dziedziną**)
 - $\cdot^{\mathcal{I}}$ jest **funkcją interpretacji** odwzorowującą
 - każdą nazwę pojęcia \mathcal{A} w podzbiór $\mathcal{A}^{\mathcal{I}}$ dziedziny $\Delta^{\mathcal{I}}$
 - każdą nazwę roli \mathcal{R} w relację binarną $\mathcal{R}^{\mathcal{I}}$ (podzbiór $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$)
 - każdą nazwę obiektu i w element $i^{\mathcal{I}}$ z $\Delta^{\mathcal{I}}$
- funkcja interpretacji rozszerza się na wyrażenia opisujące złożone pojęcia w oczywisty sposób, np. dla języka \mathcal{AL} :

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$\perp^{\mathcal{I}} = \emptyset^{\mathcal{I}}$$

$$(\neg \mathcal{A})^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus \mathcal{A}^{\mathcal{I}}$$

$$(\mathcal{C} \sqcap \mathcal{D})^{\mathcal{I}} = \mathcal{C}^{\mathcal{I}} \cap \mathcal{D}^{\mathcal{I}}$$

$$(\forall \mathcal{R}. \mathcal{C})^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b. (a, b) \in \mathcal{R}^{\mathcal{I}} \rightarrow b \in \mathcal{C}^{\mathcal{I}}\}$$

$$(\exists \mathcal{R}. \top)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b. (a, b) \in \mathcal{R}^{\mathcal{I}}\}$$

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Przykład

Założmy, że w bazie mamy następujące dane:

lostInTranslation : OscarMovie

sofiaCoppola : Director

(sofiaCoppola, lostInTranslation) : creates

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Przykład

Założmy, że w bazie mamy następujące dane:

lostInTranslation : OscarMovie

sofiaCoppola : Director

(sofiaCoppola, lostInTranslation) : creates

*Czy możemy wykazać, że: **sofiaCoppola : \forall creates.OscarMovie?***

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Przykład

Załóżmy, że w bazie mamy następujące dane:

lostInTranslation : OscarMovie

sofiaCoppola : Director

(sofiaCoppola, lostInTranslation) : creates

Czy możemy wykazać, że: sofiaCoppola : \forall creates.OscarMovie?

TAK - zamknięty świat

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Przykład

Załóżmy, że w bazie mamy następujące dane:

lostInTranslation : OscarMovie

sofiaCoppola : Director

(sofiaCoppola, lostInTranslation) : creates

Czy możemy wykazać, że: sofiaCoppola : $\forall \text{creates.OscarMovie?}$

TAK - zamknięty świat

NIE - otwarty świat

Semantyka: "świat zamknięty" kontra "świat otwarty"

- **Zamknięty świat** (bazy danych, programowanie logiczne)
 - brak informacji jest informacją negatywną (jeżeli nie możemy udowodnić prawdziwości faktu przy pomocy dostępnych danych wtedy prawdziwa jest negacja faktu)
- **Otwarty świat** (bazy wiedzy \mathcal{DL} , Semantic Web - RDF, OWL)
 - każda informacja (także negacja faktu) musi być jawnie podana

Przykład



Czy możemy wykazać, że: **sofiaCoppola** : $\forall \text{creates.OscarMovie?}$

TAK - zamknięty świat

NIE - otwarty świat

Zastosowania DL

- *Semantic Web*
- Katalogi i encyklopedie w sieci, Biblioteki cyfrowe, *Semantic Wikis*
- Medycyna (SNOMED - Systematized Nomenclature of Medicine, ok. 450 tys pojęć)
- Bioinformatyka (GeneOntology - ok. 17 tys pojęć)
- Usługi sieciowe (Web services), Zarządzanie procesami, *reasoning about actions*, transakcje w elektronicznej gospodarce (*Semantic Web services, OWL-S*)
- Przetwarzanie języka naturalnego
- Konfiguracja, integracja informacji z wielu źródeł
- Inżynieria oprogramowania, bazy danych (np. formalizowanie diagramów UML lub ER)
- Interfejs użytkownika (*Semantic Desktop*)

Semantic Web

- Pomysłodawca sieci WWW - Tim Berners-Lee, fizyk z CERN, obecnie szef W3C
- Wizja sieci: **spójna, logiczna sieć danych; dobrze zdefiniowane znaczenie informacji; współpraca pomiędzy komputerami i ludźmi = Semantic Web**

"A new form of Web content that is meaningful to computers will unleash a revolution of new abilities"

Scientific American, 2001



Sir Timothy Berners-Lee,
Przewodniczący W3C

Semantic Web



**Profesor Ian Horrocks,
University of Manchester**

Beware of the Hype!

- Hype seems to suggest that Semantic Web means: **“semantics + web = AI”**
- More realistic to think of it as meaning: **“semantics + web + AI = more useful web”**

ICCL Summer School 2005, Drezno

Semantic Web

Nota bene: Prof Ian Horrocks jest jednym z kluczowych twórców technologii i popularyzatorów idei Semantic Web



**Profesor Ian Horrocks,
University of Manchester**

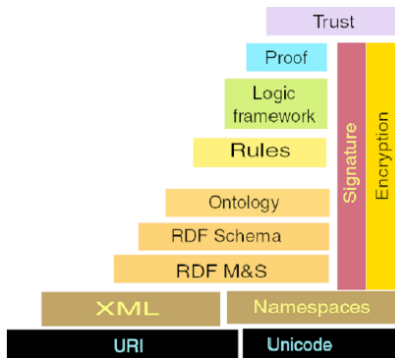
Beware of the Hype!

- Hype seems to suggest that Semantic Web means: “**semantics + web = AI**”
- More realistic to think of it as meaning: “**semantics + web + AI = more useful web**”

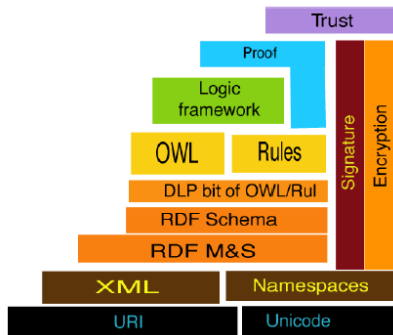
ICCL Summer School 2005, Drezno

Semantic Web - stos czy "dwie wieże"?

Początkowo zaproponowana architektura

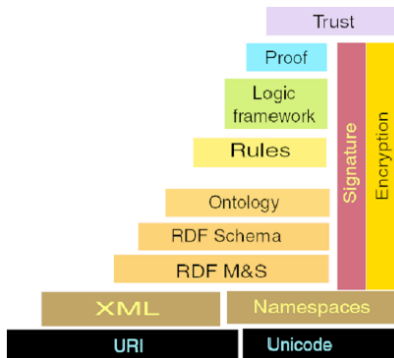


Nowa (mocno) dyskusyjna architektura (2005)

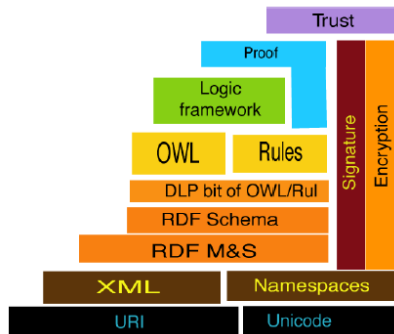


Semantic Web - stos czy "dwie wieże"?

Początkowo zaproponowana architektura



Nowa (mocno) dyskusyjna architektura (2005)



OWL (*Web Ontology Language*)

Indukcyjne programowanie logiczne

Indukcyjne programowanie logiczne

Definicja

Indukcyjne programowanie logiczne (ang. *inductive logic programming, ILP*) to dział uczenia maszynowego lub eksploracji danych, w którym wykorzystywane są techniki programowania logicznego.

Indukcyjne programowanie logiczne

Definicja

Indukcyjne programowanie logiczne (ang. *inductive logic programming, ILP*) to dział uczenia maszynowego lub eksploracji danych, w którym wykorzystywane są techniki programowania logicznego.

- cel - szukanie wzorców w danych
- reprezentacja danych i hipotez - logika pierwszego rzędu
- zaleta - łatwość włączania **wiedzy dziedzinowej**

ILP - wiedza dziedzinowa

Wiedza dziedzinowa (*ang. background/domain knowledge, BK*)
może być:

- **Ekstensjonalna** (fakty)
- **Intensjonalna** (bardziej ogólne definicje predykatów/reguły)
- w innej specjalnej postaci

Przykład wiedzy dziedzinowej

Ekstensjonalna

rodzic(ania, tomasz).
rodzic(tomasz, ewa).
kobieta(ewa).

Intensjonalna

wnuczka(X,Y):-kobieta(X), rodzic(Y,Z), rodzic(Z,X).

ILP jako przeszukiwanie

- **przestrzeń przeszukiwania**: wszystkie syntaktycznie poprawne klauzule skonstruowane z predykatów określonych przez wiedzę dziedzinową za pomocą **operatorów specjalizacji/uogólnienia** (*refinement operators*)
 - *operatory specjalizacji*: dodają literały, stosują podstawienia w ciele klauzuli ($\{X/Y\}$, $\{X/c\}$)
 - *operatory uogólnienia*: usuwają literały, stosują podstawienia odwrotne
- struktura przestrzeni hipotez opiera się na **relacji specjalizacji/uogólnienia**
- bardzo duża przestrzeń hipotez/klauzul (ograniczona przez maksymalną długość klauzuli i rozmiar wiedzy dziedzinowej)

Relacja uogólnienia

- klauzula H_1 jest **semantycznie** bardziej ogólna niż H_2 (względem teorii dziedzinowej B) wtw gdy $B \cup H_1 \models H_2$
- klauzula c_1 **θ -zawiera** c_2 ($c_1 \geq_\theta c_2$) (θ -subsumes) wtw gdy $\exists \theta : c_1 \theta \subseteq c_2$, własności:
 - odwołuje się do **syntaktyki**
 - jest poprawna (*sound*)
 - jest niezupełna (*incomplete*)
 - jest rozstrzygalna
 - jest NP-trudna

Przykład

$c_1 = p(X, Y) : \neg q(X, Y), q(Y, X)$

$c_2 = p(Z, Z) : \neg q(Z, Z)$

c_1 θ -zawiera c_2 dla podstawienia $\theta = \{X/Z, Y/Z\}$

Ograniczenie przestrzeni przeszukiwań, *declarative bias*

Declarative bias - definiowany w celu określenia i ograniczenia (dużej) przestrzeni przeszukiwań

Różne rodzaje:

- ograniczenie na język hipotez (*declarative language bias*):
dozwolone predykaty, maksymalna liczba zmiennych, typ zmiennych, zmienne wejściowe/wyjściowe
- sposób przeszukiwania (*search bias*)
- wybór strategii (*strategy bias*): przy heurystykach

ILP - deskrypcja = *Relational Data Mining*

Pierwotnie ILP predykcyjne, ostatnio zainteresowanie eksploracją danych w relacyjnych bazach danych (*Relational Data Mining, RDM*)

“odkrywanie interesujących wzorców w dużych zbiorach danych”

- **Dane.** Założenie: operowanie na pierwotnej reprezentacji (wielu tabelach), niewymagającej przetwarzania wstępnego danych do postaci jednej tabeli
- **Wzorce.** Wzorce zawierające wiele relacji zazwyczaj reprezentowane za pomocą logiki pierwszego rzędu (Datalog, Prolog)

ILP - deskrypcja = *Relational Data Mining*

Pierwotnie ILP predykcyjne, ostatnio zainteresowanie eksploracją danych w relacyjnych bazach danych (*Relational Data Mining, RDM*)

“odkrywanie interesujących wzorców w dużych zbiorach danych”

- **Dane.** Założenie: operowanie na pierwotnej reprezentacji (wielu tabelach), niewymagającej przetwarzania wstępnego danych do postaci jednej tabeli
- **Wzorce.** Wzorce zawierające wiele relacji zazwyczaj reprezentowane za pomocą logiki pierwszego rzędu (Datalog, Prolog)
- Wykorzystanie logiki obliczeniowej pozwala na włączenie do procesu uczenia wiedzy dziedzinowej np. w postaci reguł

Przykład - odkrywanie reguł asocjacyjnych w wersji RDM

WARMR (Dehaspe 1999)¹

- wzorce: relacyjne reguły asocjacyjne budowane na podstawie częstych zapytań do baz wiedzy w Prologu
- relacja uogólnienia: θ -zawieranie
- *language bias*:
 - zbiór atomów, z których można budować zapytania
 - każda zmienna w atomie oznaczona jako wyjściowa i/lub wejściowa
 - dodatkowy parametr - **atom kluczowy**

¹Dehaspe, L., Toivonen, H.: Discovery of frequent Datalog patterns. Data Mining and Knowledge Discovery, 3(1): 7 - 36, (1999)

Relacyjne częste wzorce

- zapytanie: wyrażenie logiczne $?-A_1, \dots, A_n$
- **zmienne wyróżnione** (*distinguished variables*): zmienne w *atomie kluczowym*
- **wsparcie** s zapytania to procent liczby możliwych podstawień pod zmienne wyróżnione dla danego zapytania w stosunku do liczby wszystkich możliwych podstawień pod zmienne wyróżnione

Przykład: $?- client(Key), hasCreditCard(Key, X), creditCard(X, gold)$

client
Kowalski
Nowak
Walczak

hasCreditCard	
Kowalski	12345
Kowalski	67325
Walczak	75345

creditCard	
12345	gold
67325	gold
75345	classic

wsparcie = $1/3 \approx 0.33$

Relacyjne reguły asocjacyjne

- P, Q - wzorce, $P \subseteq Q$
- Relacyjna reguła asocjacyjna - implikacja w formie $Q \rightarrow P(s, c)$
 - s - wsparcie wzorca P
 - c - zaufanie (prawdopodobieństwo, że następnik P pojawia się w zbiorze danych gdy występuje w nim poprzednik Q)

Przykład

$client(Key), hasLoan(Key, X) \Rightarrow$
 $client(Key), hasLoan(Key, X), hasCreditCard(Key, Y), creditCard(Y, gold)$
(60%, 40%)

Języki hybrydowe

Logika deskrypcyjna i programowanie logiczne

Logika deskrypcyjna

obydwa kwantyfikatory: \exists, \forall
monotoniczna negacja,
łatwość modelowania wiedzy strukturalnej

Programowanie logiczne

dowolne sterowanie zmiennymi

Logika deskrypcyjna i programowanie logiczne

Logika deskrypcyjna

obydwa kwantyfikatory: \exists, \forall
monotoniczna negacja,
łatwość modelowania wiedzy strukturalnej

Programowanie logiczne

dowolne sterowanie zmiennymi

Rozstrzygalność dzięki
aksjomatom w postaci “drzew”

+

Rozstrzygalność dzięki
skończonej liczbie instancji

Logika deskrypcyjna i programowanie logiczne

Logika deskrypcyjna

obydwa kwantyfikatory: \exists, \forall
monotoniczna negacja,
łatwość modelowania wiedzy strukturalnej

Programowanie logiczne

dowolne sterowanie zmiennymi

Rozstrzygalność dzięki
aksjomatom w postaci “drzew”

+

Rozstrzygalność dzięki
skończonej liczbie instancji

=

Nierozstrzygalność!

Description logic programs - część wspólna DL i LP

DLP = Description Logic Programs, OWL DLP, OWL Light
(WWW 2003 Grosz², Horrocks, Volz, Decker)² zdefiniowanie
ekspresyjnej części wspólnej Logiki Deskrypcyjnej i Programowania
Logicznego



² Benjamin Grosz, Ian Horrocks, Raphael Volz, and Stefan Decker, Description Logic Programs: Combining Logic Programs with Description Logics. In Proc. of WWW 2003, Budapest, Hungary, May 2003, pp. 48-57. ACM, 2003

Języki hybrydowe

- **AL-log** (Donini et al., 1998), **dl-programs** (Eiter et al., 2004): \mathcal{DL} i LP - **czarne skrzynki**, wymieniające między sobą fakty
- **CARIN** (Levy, Rousset, 1998): pojęcia i role w ciele reguł Horna w Datalogu
- **\mathcal{DL} -bezpieczne reguły** (\mathcal{DL} -safe rules) (Motik, Sattler, Studer, 2004): pojęcia i role w ciele i w głowie reguł Horna
- **SWRL** (Horrocks et al., 2004): **nierozstrzygalny**, rozszerzenie aksjomatów OWL o reguły o dowolnej postaci z kombinacjami atomów \mathcal{DL} po obu stronach reguły
 - proponowany standard w W3C
 - **nadzbior** wszystkich wymienionych powyżej podejść

Cel i zakres badań

Cel badań

Opracowanie **metod indukcyjnego programowania logicznego**,
które:

Cel badań

Opracowanie metod indukcyjnego programowania logicznego, które:

- operują na złożonych reprezentacjach, w których nacisk położony jest na semantykę (jak bazy wiedzy w logice deskrypcyjnej)

Cel badań

Opracowanie metod indukcyjnego programowania logicznego, które:

- operują na złożonych reprezentacjach, w których nacisk położony jest na semantykę (jak bazy wiedzy w logice deskrypcyjnej)
- uwzględniają wiedzę dziedzinową, reprezentowaną w ekspresyjnych, hybrydowych językach, łączących w sobie elementy Programowania logicznego i Logiki deskrypcyjnej

Cel badań

Opracowanie metod indukcyjnego programowania logicznego, które:

- operują na złożonych reprezentacjach, w których nacisk położony jest na semantykę (jak bazy wiedzy w logice deskrypcyjnej)
- uwzględniają wiedzę dziedzinową, reprezentowaną w ekspresyjnych, hybrydowych językach, łączących w sobie elementy Programowania logicznego i Logiki deskrypcyjnej
- odkrywają wzorce, które uwzględniają złożoność reprezentacji bazy wiedzy

Cel badań

Opracowanie metod indukcyjnego programowania logicznego, które:

- operują na złożonych reprezentacjach, w których nacisk położony jest na semantykę (jak bazy wiedzy w logice deskrypcyjnej)
- uwzględniają wiedzę dziedzinową, reprezentowaną w ekspresyjnych, hybrydowych językach, łączących w sobie elementy Programowania logicznego i Logiki deskrypcyjnej
- odkrywają wzorce, które uwzględniają złożoność reprezentacji bazy wiedzy
- korzystają z relacji specjalizacji/uogólnienia, która w pełni uwzględnia semantykę reprezentacji (zamiast po części opierającej się na syntaktyce jak θ -zawieranie)

Wiedza dziedzinowa

- badania zaczynamy od najmniejszego podzbioru \mathcal{DL} i LP -
Description Logic Programs
- następnie będziemy rozważać bardziej złożone języki w
ramach podejścia **\mathcal{DL} -bezpieczne reguły**

Wiedza dziedzinowa - \mathcal{DL} -bezpieczne reguły³

\mathcal{KB} - baza wiedzy w logice deskrypcyjnej

\mathcal{P} - zbiór reguł

\mathcal{DL} -bezpieczne reguły

Reguła r nazywana **\mathcal{DL} -bezpieczną**, jeżeli każda zmienna występująca w r pojawia się w ciele reguły w jakimś atomie, spoza zestawu symboli \mathcal{KB}

- Semantyka łączonej bazy wiedzy $(\mathcal{KB}, \mathcal{P})$ dana poprzez translację do logiki pierwszego rzędu jako $\pi(\mathcal{KB}) \cup \mathcal{P}$
- Główny rodzaj wnioskowania w $(\mathcal{KB}, \mathcal{P})$: odpowiadanie na zapytania (*query answering*), tzn. decydowanie czy $\pi(\mathcal{KB}) \cup \mathcal{P} \models \alpha$ dla faktu α .

³

Motik B., Sattler U., Studer R. Query Answering for OWL-DL with Rules. Proc. of the 3rd International Semantic Web Conference (ISWC 2004), Hiroshima, Japan, November, 2004

Wiedza dziedzinowa - \mathcal{DL} -"bezpieczeństwo"

- Pojęcie podobne do "bezpieczeństwa" w Datalogu
 - reguła bezpieczna - każda zmienna pojawia się w pozytywnym literale w ciele reguły (powoduje to jej wiązanie z wartościami obecnymi w bazie)
 - \mathcal{DL} -"bezpieczeństwo" działa podobnie: każda zmienna wiązana tylko z instancjami obecnymi w ABoxie

Przykład - (\mathcal{KB} zawiera **Homeworker**, **livesAt**, **Person**, **worksAt**)

$\text{Homeworker}(x) \leftarrow \text{Person}(x), \text{livesAt}(x, y), \text{worksAt}(x, y)$
(reguła nie jest \mathcal{DL} -bezpieczna, ponieważ x i y występują tylko w atomach \mathcal{DL})

$\text{Homeworker}(x) \leftarrow \text{Person}(x), \text{livesAt}(x, y), \text{worksAt}(x, y), \mathcal{O}(x), \mathcal{O}(y)$
(zakładamy, że istnieje fakt $\mathcal{O}(\alpha)$ dla każdej instancji α w ABoxie)

Definicja zadania odkrywania częstych wzorców w bazach OWL DLP⁴

Mając daną:

- **bazę wiedzy** \mathcal{KB} reprezentowaną w języku OWL DLP, zawierającą \mathcal{DL} -bezpieczne reguły,
- **zbiór wzorców** w języku \mathcal{L} w postaci zapytań Q , które wszystkie zawierają **pojęcie referencyjne** C_{ref} ,
- **minimalny próg wsparcia** $minsup$, podany przez użytkownika i zakładając, że zapytania ze wsparciem s są częste w \mathcal{KB} mając dane C_{ref} , jeżeli $s \geq minsup$)

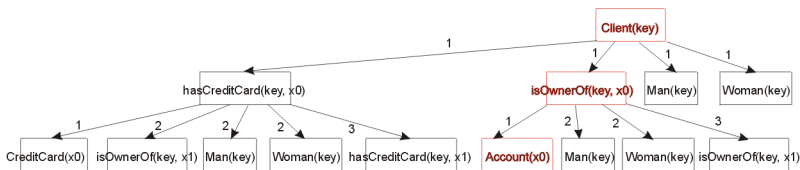
zadanie odkrywania częstych wzorców polega na **znalezieniu** zbioru \mathcal{F} częstych zapytań.

Wsparcie s oblicza się względem **pojęcia referencyjnego**, analogicznie jak to było omówione w przypadku RDM względem atomu kluczowego.

⁴ Józefowska J., Ławrynowicz A., Łukaszewski T. Towards discovery of frequent patterns in description logics with rules, RuleML 2005 (Rules and Rule Markup Languages for the Semantic Web), Galway, Ireland, 2005, LNCS, Springer Verlag

Algorytm⁵

- bazuje na specjalnej strukturze (*trie*) (FARMER)
- węzły reprezentują atomy zapytania
- ścieżka od korzenia do liścia reprezentuje całe zapytanie
- węzły dodawane do drzewa na trzy sposoby (zaznaczone na rysunku):
 - 1 jako węzły zależne (podpojęcia, role wychodzące z pojęć itp.)
 - 2 jako kopie prawych braci węzła
 - 3 jako kopie samego węzła
- dodatkowo sprawdzane ograniczenia (role funkcyjne, odwrotne itp.)



q(key):-Client(key), isOwnerOf(key, x0), Account(x0)

⁵ Józefowska J., Ławrynowicz A., Łukaszewski T. Faster frequent pattern mining from the Semantic Web, IIS: IIPWM'06, Advances in Soft Computing, Springer Verlag 2006, accepted for publication

Implementacja

- Dane testowe
 - *problem*: brak dostępnych ontologii z dużym ABox'em
 - *zdefiniowaliśmy ontologię* na podstawie relacyjnych danych z PKDD Discovery Challenge'99 (usługi bankowe - udostępniona online⁶)
- Implementacja: w Javie
- Używany system do obsługi bazy wiedzy: **KAON2**⁷ - OWL, *DL*-bezpieczne reguły
 - ostatnio wysłane do publikacji testy (m.in. na naszej ontologii): gdy mały TBox i duży ABox - nawet kilkakrotnie szybszy od dwóch konkurencyjnych jak RACER i Pellet

⁶ <http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm>

⁷ <http://kaon2.semanticweb.org>

Przykładowy wynik zastosowania algorytmu

Częsty wzorzec (min wsparcie: 20%, pojęcie referencyjne: *Gold*)

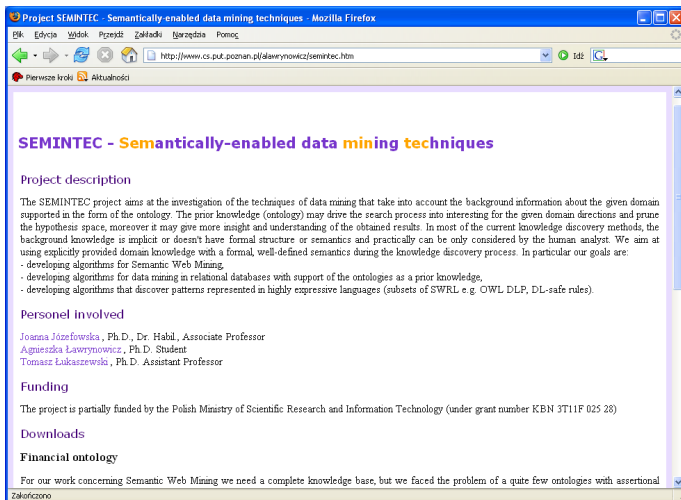
$q(Key)$: —

$hasCreditCard(X_1, Key)$, $hasSexValue(X_1, X_2)$, $MaleSex(X_2)$,
 $hasAgeValue(X_1, X_3)$, $From50To65(X_3)$, $isOwnerOf(X_1, X_4)$,
 $hasOwner(X_4, X_1)$, $Account(X_4)$, $livesIn(X_1, X_5)$, $Region(X_5)$,
 $Client(X_1)$, $Gold(Key)$ (wsparcie: 21,59%)

“mężczyzna w wieku pomiędzy 50-65 lat, posiadacz konta i złotej karty kredytowej”

SEMINTEC

Strona projektu **SEMINTEC** (opis, dane testowe, lista publikacji itd.)
<http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm>



The screenshot shows a Mozilla Firefox browser window with the title "Project SEMINTEC - Semantically-enabled data mining techniques - Mozilla Firefox". The address bar shows the URL "http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm". The page content includes:

SEMINTEC - Semantically-enabled data mining techniques

Project description

The SEMINTEC project aims at the investigation of the techniques of data mining that take into account the background information about the given domain supported in the form of the ontology. The prior knowledge (ontology) may drive the search process into interesting for the given domain directions and prune the hypothesis space, moreover it may give more insight and understanding of the obtained results. In most of the current knowledge discovery methods, the background knowledge is implicit or doesn't have formal structure or semantics and practically can be only considered by the human analyst. We aim at using explicitly provided domain knowledge with a formal, well-defined semantics during the knowledge discovery process. In particular our goals are:

- developing algorithms for Semantic Web Mining,
- developing algorithms for data mining in relational databases with support of the ontologies as a prior knowledge,
- developing algorithms that discover patterns represented in highly expressive languages (subsets of SWRL e.g. OWL DLP, DL-safe rules).

Personel involved

Joanna Józefowska, Ph.D., Dr. Habil., Associate Professor
Agnieszka Ławrynowicz, Ph.D. Student
Tomasz Łukaszewski, Ph.D. Assistant Professor

Funding

The project is partially funded by the Polish Ministry of Scientific Research and Information Technology (under grant number KBN 3T11F 025 28)

Downloads

Financial ontology

For our work concerning Semantic Web Mining we need a complete knowledge base, but we faced the problem of a quite few ontologies with assertional

Zakończono

Aktualny stan badań w dziedzinie

- Relacyjna eksploracja danych
 - algorytmy **WARMR** (Dehaspe, Toivonen, 1999, , Blockeel et al., 2002) i **FARMER** (Nijssen, Kok, 2001, 2003)
- Bardziej złożone reprezentacje
 - **SPADA** (Lisi, Malerba, 2004) jednak komponent \mathcal{DL} jak na razie mocno ograniczony w tym podejściu

Aktualny stan badań w dziedzinie - dyskusja

- generalny problem - duża przestrzeń przeszukiwań
- **Metody RDM**
 - najczęściej używana relacja θ -zawieranie jest tylko przybliżeniem logicznej konsekwencji
 - problem z wiedzą strukturalną/hierarchiczną
- **SPADA**
 - działa na ograniczonej wersji języka \mathcal{AL} -log
 - zakłada odkrywanie reguł na wielu poziomach szczegółowości (wykorzystuje taksonomie)
 - obecna wersja nie wykorzystuje złożonych pojęć ani ról
 - albo ograniczamy się do odkrywania wzorców, które zawierają tylko pojęcia z tego samego poziomu hierarchii, albo musimy pojęcia replikować na wielu poziomach hierarchii

Kierunki dalszych prac

- opracowanie technik zwiększających efektywność algorytmu dokładnego
- rozpatrzenie coraz bardziej zaawansowanych języków w ramach podejścia \mathcal{DL} -bezpieczne reguły
- opracowanie heurystyk
- opracowanie algorytmów dla innych metod odkrywania wiedzy
- opracowanie algorytmów przy założeniu *UNA/OI* (*Unique Name Assumption/Object Identity*) i wykorzystaniu niemonotonicznych rozszerzeń języka OWL

Kierunki dalszych prac

- **opracowanie technik zwiększających efektywność algorytmu dokładnego**
- **rozpatrzenie coraz bardziej zaawansowanych języków w ramach podejścia \mathcal{DL} -bezpieczne reguły**
- **opracowanie heurystyk**
- **opracowanie algorytmów dla innych metod odkrywania wiedzy**
- **opracowanie algorytmów przy założeniu *UNA/OI* (*Unique Name Assumption/Object Identity*) i wykorzystaniu niemonotonicznych rozszerzeń języka OWL**

Dziękuję za uwagę!

Literatura 1/2



Horrocks I., Parsia B., Patel-Schneider P. F., Hendler J. A., Semantic Web Architecture: Stack or Two Towers?, PPSWR 2005



Grosof B., Horrocks I., Volz R., Decker S., Description Logic Programs: Combining Logic Programs with Description Logics. In Proc. of WWW 2003, Hungary, ACM, 2003



Donini F. M., Lenzerini M., Nardi D., Schaerf A., AL-log: Integrating Datalog and Description Logics. J. Intell. Inf. Syst. 10(3): 227-252 (1998)



Eiter T., Lukasiewicz T., Schindlauer R., Tompits H., Combining answer set programming with description logics for the semantic web. In Proc. of the International Conference of Knowledge Representation and Reasoning (KR04), 2004



Levy, A., Rousset, M.-C. (1998). Combining Horn rules and description logics in CARIN. Artificial Intelligence, 104, 165–209








Motik B., Sattler U., Studer R. Query Answering for OWL-DL with Rules. Proc. of the 3rd International Semantic Web Conference (ISWC 2004), Japan, 2004



Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M., SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission 21 May 2004, <http://www.w3.org/Submission/SWRL/>

Literatura 2/2

-  Józefowska J., Ławrynowicz A., Łukaszewski T. Towards discovery of frequent patterns in description logics with rules, RuleML 2005 (Rules and Rule Markup Languages for the Semantic Web), Galway, Ireland, 2005, LNCS, Springer Verlag
-  Józefowska J., Ławrynowicz A., Łukaszewski T. Faster frequent pattern mining from the Semantic Web, IIS: IIPWM'06, Advances in Soft Computing, Springer Verlag 2006, accepted for publication
-  Dehaspe, L., Toivonen, H.: Discovery of frequent Datalog patterns. Data Mining and Knowledge Discovery, 3(1): 7 - 36, (1999)
-  Nijssen, S., Kok, J.N. Faster Association Rules for Multiple Relations, IJCAI'01
-  Nijssen, S. Kok, J.N.. Efficient frequent query discovery in FARMER, PKDD 2003, LNAI, pp 350-362, Springer-Verlag, 2003
-  Lisi F.A., Malerba D., Inducing Multi-Level Association Rules from Multiple Relation, Machine Learning Journal, 55, 175-210, (2004)
-  Motik B., Sattler U. Practical DL Reasoning over Large ABoxes with KAON2. Submitted for publication

Interpretacja

$$\begin{aligned} & (Artist \sqcap \exists creates.Movie)^{\mathcal{I}} \\ &= \\ & (Artist)^{\mathcal{I}} \cap (\exists creates.Movie)^{\mathcal{I}} \\ &= \\ & \{x \mid Artist(x)\} \cap \\ & \{x \mid \exists y. creates(x, y) \wedge Movie(y)\} \\ &= \\ & \{x \mid Artist(x) \wedge \\ & \exists y. creates(x, y) \wedge Movie(y)\} \end{aligned}$$

ILP - predykcja

W ujęciu klasycznym (predykcyjnym) ILP definiuje się następująco:

ILP - predykcja

W ujęciu klasycznym (predykcyjnym) ILP definiuje się następująco:

Mając dane:

- Wiedzę dziedzinową B (w postaci zbioru klauzul)
- Zbiór przykładów *pozytywnych* E^+
- Zbiór przykładów *negatywnych* E^-
- *Language bias* \mathcal{L} - ograniczenia na format nowych klauzul

ILP - predykcja

W ujęciu klasycznym (predykcyjnym) ILP definiuje się następująco:

Mając dane:

- Wiedzę dziedzinową B (w postaci zbioru klauzul)
- Zbiór przykładów *pozytywnych* E^+
- Zbiór przykładów *negatywnych* E^-
- *Language bias* \mathcal{L} - ograniczenia na format nowych klauzul

Znajdź hipotezę H (zbiór relacji wyrażony jako program logiczny - nowe klauzule) taką, że:

- spełnia ona ograniczenia języka
- zbiór E^+ nie jest logiczną konsekwencją B
- wszystkie elementy E^+ są logiczną konsekwencją $B \wedge H$
($B \wedge H \models E^+$) - **completeness**
- i nie jest nią żaden element E^- - **consistency**

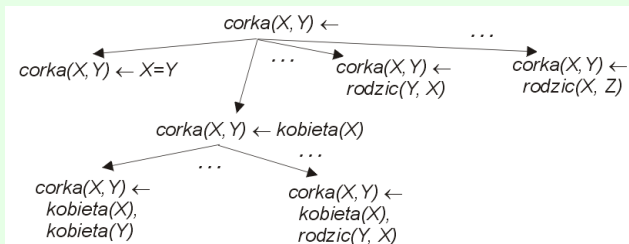
ILP - kluczowe pojęcia

- wiedza dziedzinowa
- ILP jako przeszukiwanie: operatory specjalizacji/uogólnienia
- ograniczenie przestrzeni przeszukiwań: "declarative bias"

ILP jako przeszukiwanie - przykład

Przykład

Przykłady uczące	Wiedza dziedzinowa (ekstensjonalna)	
$\text{corka}(\text{maria}, \text{ania}). \oplus$	$\text{rodzic}(\text{ania}, \text{maria}).$	$\text{kobieta}(\text{ania}).$
$\text{corka}(\text{ewa}, \text{tomasz}). \oplus$	$\text{rodzic}(\text{ania}, \text{tomasz}).$	$\text{kobieta}(\text{maria}).$
$\text{corka}(\text{tomasz}, \text{ania}). \ominus$	$\text{rodzic}(\text{tomasz}, \text{ewa}).$	$\text{kobieta}(\text{ewa}).$
$\text{corka}(\text{ewa}, \text{ania}). \ominus$	$\text{rodzic}(\text{tomasz}, \text{jan}).$	



ILP jako przeszukiwanie: uogólnienie/specjalizacja

- struktura przestrzeni hipotez opiera się na relacji uogólnienia/specjalizacji
 - Niech C i D będą dwoma klauzulami. C jest bardziej ogólna niż D wtedy i tylko wtedy gdy $C \models D$, tzn., gdy $\text{pokrywa}(D) \subseteq \text{pokrywa}(C)$
- ograniczanie przestrzeni
 - jeżeli H nie pokrywa przykładu e wtedy żadna specjalizacja nie będzie pokrywać e

Operatory uogólnienia/specjalizacji (*refinement operators*)

- generują graf specjalizacji/uogólnienia
- **operatory specjalizacji:**
 - dodają literały do ciała klauzuli
 - stosują podstawienia w ciele klauzuli np. $\{X/Y\}$ gdzie X, Y znajdowały się już w atomie, $\{X/c\}$ gdzie c to stała
- **operatory uogólnienia:**
 - usuwają literały z ciała klauzuli
 - stosują podstawienia odwrotne

Operator idealny

Operator idealny

- wszystkie specjalizacje/uogólnienia generowane w skończonym czasie (*local finiteness*),
 - możliwe uzyskanie wszystkich klauzul w danym języku, bardziej szczegółowych/ogólnych od danej klazuli, zgodnie z przyjętą miarą ogólności (*completeness*),
 - uzyskane klauzule zawsze bardziej szczegółowe/ogólne od danej klauzuli, nigdy równoważne (*properness*)
-
- dla klauzul Horna jako języka reprezentacji i θ -zawierania jako miary ogólności niemożliwe jest zdefiniowanie idealnego operatora
 - niektóre klauzule mają nieskończoną liczbę poprawnych, minimalnych specjalizacji
 - istnienie nieskończonych łańcuchów
$$:-p(X1,X2),p(X2,X1), p(X1,X3),p(X3,X1),p(X2,X3),p(X3,X2)$$

Przykład - odkrywanie reguł asocjacyjnych w wersji RDM

WARMR (Dehaspe 1998)⁸

- wzorce: relacyjne reguły asocjacyjne są budowane na podstawie częstych zapytań do baz wiedzy w Prologu
- miara generalizacji: θ -zawieranie
- *language bias*: WARMODE (deklaracja języka zapytań \mathcal{L})
 - zbiór atomów, z których budujemy zapytania
 - każda zmienna w atomie oznaczona jest jako :
 - - wyjściowa, + wejściowa, -+wyjściowo/wejściowa
 - dodatkowy parametr - **atom kluczowy** (określa co jest liczone)
 - typy zmiennych i dodatkowe ograniczenia na postać wzorców

Klucz	Atomy	Przykład wzorca $\in \mathcal{L}$	Przykład wzorca $\notin \mathcal{L}$
client(-)	hasCreditCard(+,-), hasLoan(+,-)	client(X), hasCreditCard(X,Y), hasLoan(X,Z)	client(X), hasCreditCard(Y,X)

⁸Dehaspe, L., Toivonen, H.: Discovery of frequent Datalog patterns. Data Mining and Knowledge Discovery, 3(1): 7 - 36, (1999)

Logika deskrypcyjna i programowanie logiczne

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne
- Każda z tych logik posiada pewne własności, których z kolei nie posiada druga

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne
- Każda z tych logik posiada pewne własności, których z kolei nie posiada druga
 - **Logika deskrypcyjna** - ograniczenie do reguł w postaci "drzew", ale możliwość definiowania zmiennych związanych kwantyfikatorem ogólnym bądź egzystencjalnym i monotoniczna negacja; łatwość modelowania wiedzy strukturalnej/hierarchicznej

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne
- Każda z tych logik posiada pewne własności, których z kolei nie posiada druga
 - **Logika deskrypcyjna** - ograniczenie do reguł w postaci "drzew", ale możliwość definiowania zmiennych związanych kwantyfikatorem ogólnym bądź egzystencjalnym i monotoniczna negacja; łatwość modelowania wiedzy strukturalnej/hierarchicznej
 - **Programowanie logiczne** - ograniczenie do kwantyfikatora ogólnego, ale za to możliwość dowolnego sterowania zmiennymi

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne
- Każda z tych logik posiada pewne własności, których z kolei nie posiada druga
 - **Logika deskrypcyjna** - ograniczenie do reguł w postaci "drzew", ale możliwość definiowania zmiennych związanych kwantyfikatorem ogólnym bądź egzystencjalnym i monotoniczna negacja; łatwość modelowania wiedzy strukturalnej/hierarchicznej
 - **Programowanie logiczne** - ograniczenie do kwantyfikatora ogólnego, ale za to możliwość dowolnego sterowania zmiennymi
- Pożądane połączenie ekspresyjności tych dwóch logik

Logika deskrypcyjna i programowanie logiczne

- Logika deskrypcyjna (OWL DL) i Programowanie logiczne (reguły ograniczone do klauzul Horna) - logiki rozstrzygalne
- Każda z tych logik posiada pewne własności, których z kolei nie posiada druga
 - **Logika deskrypcyjna** - ograniczenie do reguł w postaci "drzew", ale możliwość definiowania zmiennych związanych kwantyfikatorem ogólnym bądź egzystencjalnym i monotoniczna negacja; łatwość modelowania wiedzy strukturalnej/hierarchicznej
 - **Programowanie logiczne** - ograniczenie do kwantyfikatora ogólnego, ale za to możliwość dowolnego sterowania zmiennymi
- Pożądane połączenie ekspresyjności tych dwóch logik
- Ostatnio - duży wzrost zainteresowania tym problemem, niestety taka kombinacja w ogólności prowadzi do **nierozstrzygalności**

Języki hybrydowe

- **AL-log** (Donini et al., 1998)⁹, **dl-programs** (Eiter et al., 2004)¹⁰:
DL i LP - *czarne skrzynki*, wymieniające między sobą fakty
 - w ciele reguł LP zapytania do ontologii w DL, rozpatrywane podczas ewaluacji reguł LP. Fakty mogą zostać wysłane do ontologii DL przed wykonaniem zapytania.
 - **AL-log**: dodatkowe ograniczenia (pojęcia DL) w regułach w Datalogu, określające typ zmiennych w ciele reguły
 - **dl-programs**: LP rozszerzone o negację pod semantyką modeli stabilnych (*Stable Model Semantics*) lub semantyką dobrze ufundowaną (*Well-Founded Semantics*), specjalne predykaty w ciele reguł LP, zawierające zapytania o fakty (wynik wnioskowania) do bazy wiedzy DL
- **CARIN** (Levy, Rousset, 1998)¹¹, Pojęcia i role w ciele reguł Horna w Datalogu

⁹ Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Andrea Schaerf: AL-log: Integrating Datalog and Description Logics. J. Intell. Inf. Syst. 10(3): 227-252 (1998)

¹⁰ Thomas Eiter, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits. Combining answer set programming with description logics for the semantic web. In Proc. of the International Conference of Knowledge Representation and Reasoning (KR04), 2004

¹¹ Levy, A., Rousset, M.-C. (1998). Combining Horn rules and description logics in CARIN. Artificial Intelligence, 104, 165–209

Języki hybrydowe c.d.

- **DL-bezpieczne reguły** (*DL-safe rules*) (ISWC 2004, Motik, Sattler, Studer)¹² - rozstrzygalne; pojęcia i role mogą pojawiać się w ciele i w głowie reguł Horna.
 - DL-bezpieczność: każda zmienna jest wiązana z instancjami, które są jawnie umieszczone w ABoxie
- **SWRL**¹³ - **nierozstrzygalny**, rozszerzenie aksjomatów OWL o reguły o dosyć dowolnej postaci, w których występują kombinacje atomów DL po obu stronach reguły
 - SWRL stał się częścią proponowanych standardów w W3C
 - SWRL jest **nadzbiorem** wszystkich wymienionych powyżej podejść

¹² Motik B., Sattler U., Studer R. Query Answering for OWL-DL with Rules. Proc. of the 3rd International Semantic Web Conference (ISWC 2004), Hiroshima, Japan, November, 2004

¹³ Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M., SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission 21 May 2004, <http://www.w3.org/Submission/SWRL/>

Wzorce - zapytania koniunkcyjne

- **Zapytanie koniunkcyjne** $Q(\mathbf{x}, \mathbf{y})$ do (\mathcal{KB}, P) - koniunkcja atomów o formie $P(s_1, \dots, s_n)$ i zawierająca dokładnie zbiory zmiennych \mathbf{x} i \mathbf{y} , gdzie:
 - \mathbf{x} - zmienne wyróżnione, \mathbf{y} - zmienne pomocnicze
 - P jest symbolem predykatu
 - s_1, \dots, s_n instancjami z (\mathcal{KB}, P) lub zmiennymi
- Odpowiedź na zapytanie $Q(\mathbf{x}, \mathbf{y})$ względem (\mathcal{KB}, P) jest przypisaniem instancji do zmiennych wyróżnionych, takim że $\pi(\mathcal{KB}) \models \pi Q(\mathbf{x}\theta, \mathbf{y})$

Przykład

Dla przejrzystości używamy następującego zapisu na oznaczenie zapytań:
 $q(key) : - C_{ref}(key), P_1(s_1, \dots, s_{m1}), \dots, P_n(k_1, \dots, k_{mn})$ gdzie:

- $q(key)$ oznacza, że key to jedyna zmienna wyróżniona w zapytaniu
- $P_1(s_1, \dots, s_{m1}), \dots, P_n(k_1, \dots, k_{mn})$ reprezentują atomy zapytania
- C_{ref} to **pojęcie referencyjne** (rola analogiczna do roli atomu kluczowego)

$q(key) : - Client(key), isOwnerOf(key, x), Account(x), O(key), O(x)$